AMENDMENTS TO THE CLAIMS

The following listing of claims will replace all prior versions, and listings, of claims in the application:

1. (Currently Amended) A multiprocessor system comprising a first processor and a second processor, wherein:

the first processor comprises an interrupt generation unit which generates an interrupt to the second processor when the first processor executes a predetermined call instruction in a running main routine, the interrupt being for requesting to store a return address for returning to the main routine upon completion of processing of a subroutine called by the call instruction; and

the second processor comprises an address save unit which saves ~~a~~ the return address ~~for returning to the main routine upon completion of processing of a subroutine called by the call instruction~~ to a predetermined memory area when the second processor receives an interrupt from the interrupt generation unit.

2. (Currently Amended) The multiprocessor system according to claim 1, wherein:

the interrupt generation unit generates an interrupt to the second processor again when a predetermined return instruction is executed in the subroutine; and

the second processor further comprises an address notification unit which ~~notifies~~ communicates the return address to the first processor when receiving the re-generated interrupt.

3. (Previously Presented) The multiprocessor system according to claim 1, wherein

the first processor comprises a fetcher which fetches an instruction; and the return address is set as a target address to be accessed by the fetcher.

4.(Currently Amended) A multiprocessor system comprising a first processor and a second processor, wherein:

the first processor comprises an interrupt generation unit which generates an interrupt to the second processor when the first processor executes a predetermined call instruction or a jump instruction, the interrupt being for requesting to extract a call destination address or a jump destination address stored dividedly in formats of the call instruction or the jump instruction and an accompanying execution stop instruction; and

the second processor comprises:

an address extraction unit which extracts ~~a~~ the call destination address or the jump destination address ~~stored dividedly in formats of the call instruction or the jump instruction and an accompanying execution stop instruction~~ when the second processor receives the interrupt from the first processor; and

an address notification unit which communicates the acquired call destination address or jump destination address to the first processor.

5.(Previously Presented) The multiprocessor system according to claim 4, wherein:

the first processor further comprises a fetcher which fetches an instruction; and

the call destination address or the jump destination address is set as a target address to be accessed by the fetcher.

6. (Currently Amended) A multiprocessor system comprising a graphics processor and a main processor, wherein

the graphics processor comprises:

a direct memory access controller (DMAC) which reads instructions written in a display list from a memory sequentially;

a decoder which decodes the read instructions sequentially; and

an interrupt generation unit which generates a shift interrupt to the main processor when a decoded instruction is a predetermined call instruction included in a main routine of the display list and generates a return interrupt to the main processor when a decoded instruction is a return instruction included in a subroutine called by the call instruction, the shift interrupt being for requesting to store a return address for returning to the main routine upon completion of processing of the subroutine, and the return interrupt being for requesting to transfer the return address,

the main processor comprises:

an address save unit which saves ~~a~~ the return address ~~for returning to the main routine upon completion of processing of the subroutine~~ to a predetermined memory when the main processor receives the shift interrupt from the interrupt generation unit; and

an address notification unit which reads the return address from the predetermined memory and communicates the return address to the graphics processor when the main processor receives the return interrupt from the interrupt generation unit, and

the return address communicated to the graphics processor is set as a target address to be accessed by the DMAC.

7.(Currently Amended) A multiprocessor system comprising a graphics processor and a main processor, wherein

the graphics processor comprises:

a direct memory access controller (DMAC)which reads instructions written in a display list from a memory sequentially;

a decoder which decodes the read instructions sequentially; and

an interrupt generation unit which generates an interrupt to the main processor when a decoded instruction is a predetermined call instruction or a jump instruction included in the display list, the interrupt being for requesting to extract a call destination address or a jump destination address stored dividedly in formats of the call instruction or the jump instruction and an accompanying execution stop instruction,

the main processor comprises an address notification unit which acquires ~~a~~ the call destination address or ~~a~~ the jump destination address ~~stored dividedly in formats of the call instruction or the jump instruction and an accompanying execution stop instruction~~, and communicates the call destination address or the jump destination address to the graphics processor when the main processor receives the interrupt from the interrupt generation unit, and

the call destination address or the jump destination address communicated to the graphics processor is set as a target address to be accessed by the DMAC.

8.(Currently Amended) A method of executing a program in a multiprocessor system, the method comprising, executing a call instruction in a main routine running by a first processor, generating an interrupt for delegating to a second processor the task of saving a return address

for returning to the main routine upon completion of processing of a subroutine called by the call instruction.

9.(Previously Presented) The method of executing a program in a multiprocessor system according to claim 8, wherein:

if a stack area inside the first processor has a free space, the first processor saves the return address to the stack area by itself; and

if the stack area has no free space, the save of the return address is delegated to the second processor.

10.(Previously Presented) The method of executing a program in a multiprocessor system according to claim 8, wherein:

if the call instruction does not explicitly instruct to delegate the task of saving the return address to the second processor, the first processor saves the return address to a stack area built in itself; and

if the call instruction explicitly instructs to delegate the task of saving the return address to the second processor, the first processor delegates the task of saving the return address to the second processor.

11. (Currently Amended) A method of executing a program in a multiprocessor system, the method comprising, executing a call instruction or a jump instruction by a first processor, generating an interrupt for delegating a task of acquiring a full address of a call destination

address or a jump destination address stored dividedly in formats of the call instruction or the jump instruction and an accompanying execution stop instruction to a second processor.

12. (Previously Presented) A method of executing a program in a multiprocessor system according to claim 11, wherein delegating a task of acquiring the full address of the call destination address or the jump destination address to a second processor, if the number of bits of the call destination address or the jump destination address exceeds the number of bits acquirable by the first processor.

13. (Canceled)

14. (Previously Presented) A method of executing a program in a multiprocessor system according to claim 11, wherein delegating the task of acquiring the full address of the call destination address or the jump destination address to a second processor if the call instruction or the jump instruction explicitly instructs to delegate the task of acquiring the call destination address or the jump destination address to the second processor.